EasyGui 学习文档

翻译改编者　小甲鱼
排版　天涯客
本教程出自鱼 C 工作室
(本文档由 LaTeX 排版系统编辑而成, 编辑环境:TeXLive 2014 + TeXmaker)

2015-01-13

Contents

1 安装 EasyGui

官网
最新版:easygui-0.97.zip
使用标准方法安装:

1. 使用命令窗口切换到`easygui-docs-0.96` 的目录下
2. Windows 下执行`C:\Python33\python.exe setup.py install`
3. Linux 或 Mac 下`sudo /usr/bin/python33 setup.py install`

2 建议不要在 IDLE 上运行 EasyGui

EasyGui 是运行在 Tkinter 上并拥有自身的事件循环, 而 IDLE 也是 Tkinter 写的一个应用程序并也拥有自身的事件循环. 因此当两者同时运行的时候, 有可能会发生冲突, 且带来不可预测的结果. 因此如果你发现你的 EasyGui 程序有这样的问题, 请尝试在 IDLE 外去运行你的程序.

3 一个简单的例子

在 EasyGui 中, 所有的 GUI 互动均是通过简单的函数调用, 下边一个简单的例子告诉你 EasyGui 确实很 Easy!

```
1  import easygui as g
2  import sys
3
4  while 1:
5          g.msgbox("嗨，欢迎进入第一个界面小游戏^_^")
6
7          msg ="请问你希望在鱼C工作室学习到什么知识呢？"
8          title  = "小游戏互动"
9          choices = ["谈恋爱", "编程", "OOXX", "琴棋书画"]
10
11         choice = g.choicebox(msg, title,  choices)
12
13         # note that we convert choice to string,  in  case
14         # the user cancelled the choice,  and we got None.
15         g.msgbox("你的选择是:" + str(choice), "结果")
```

```
16
17          msg = "你希望重新开始小游戏吗？"
18          title  = "请选择"
19
20          if g.ccbox(msg, title):     #     show a Continue/Cancel dialog
21                  pass # user chose Continue
22          else:
23                  sys.exit(0)  #     user chose Cancel
```

4 EasyGui 的各种功能演示

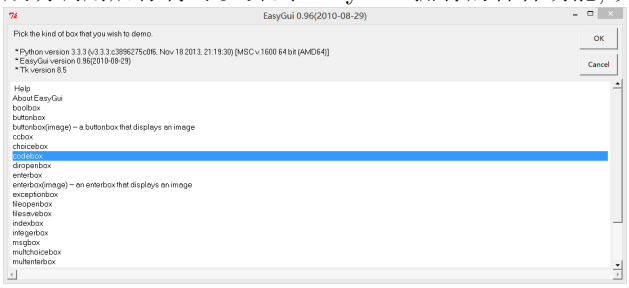
要运行 EasyGui 的演示程序, 在命令行调用 EasyGui 是这样的:

```
1 C:\Python33\python.exe easygui.py
```

或者你可以从 IDE(例如 IDLE,PythonWin,Wing, 等等) 上来调用:

```
1 >>> import easygui as g
2 >>> g.egdemo()
```

成功调用后你将可以尝试 EasyGui 拥有的各种功能, 并将你选择的结果打印至控制台.



5 导入 EasyGui

为了使用 EasyGui 这个模, 你应该先导入它. 最简单的导入语句是:

```
1 import easygui
```

如果你使用上面这种形式导入的话, 那么你使用 EasyGui 的函数的时候, 必须在函数的前面加上前缀 easygui, 像这样:

```
easygui.msgbox(...)
```

另一种选择是导入整个 EasyGui 包:

```
from easygui import *
```

这使得我们更容易调用 EasyGui 的函数, 你可以直接这样编写代码:

```
msgbox(...)
```

第三种方案是使用类似下边的 import 语句:

```
import easygui as g
```

这样可以让你保持 EasyGui 的命名空间, 同时减少你的打字数量. 导入之后你就可以这么调用 EasyGui 的函数:

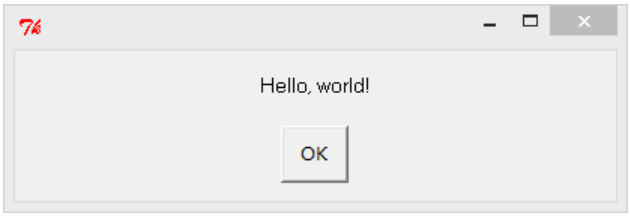
```
g.msgbox(...)
```

6 使用 EasyGui

一旦你的模块导入 EasyGui,GUI 操作就是一个简单的调用 EasyGui 函数的几个参数的问题了.

例如, 使用 EasyGui 来实现著名的 “你好, 世界!” 程序是这样的:

```
import easygui as g
    g.msgbox("Hello, world!")
```



7 EasyGui 函数的默认参数

对于所有函数而言, 前两个参数是消息和标题. 按照这个规律, 在某种情况下, 这可能不是最有利于用户的安排 (例如, 对话框在获取目录和文件名的时候忽略消息参数), 但我觉得保持这种一致性贯穿于所有的窗口部件是一种更为重要的考虑!

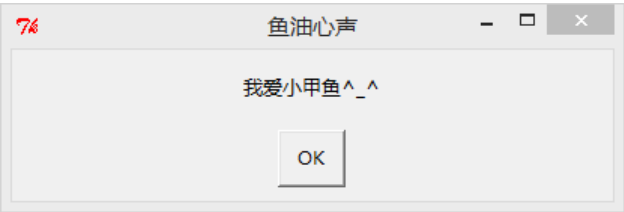
绝大部分的 EasyGui 函数都有默认参数, 几乎所有的组件都会显示一个消息和标题. 标题默认是空字符串, 信息通常有一个简单的默认值.

这使得你可以尽量少的去设置参数, 比如 `msgbox()` 函数标题部分的参数就是可选的, 所以你调用 `msgbox()` 的时候可以只指定一个消息参数, 例如:

```
>>> msgbox('我爱小甲鱼^_^')
```

当然你也可以指定标题参数和消息参数, 例如:

```
>>> msgbox('我爱小甲鱼^_^', '鱼油心声')
```



在各类按钮组件里, 默认的消息是 "Shall I continue?", 所以你可以不带任何参数地去调用它们. 这里我们演示不带任何参数地去调用 `ccbox()`, 当选择 "cancel" 或关闭窗口的时候返回一个布尔类型的值:

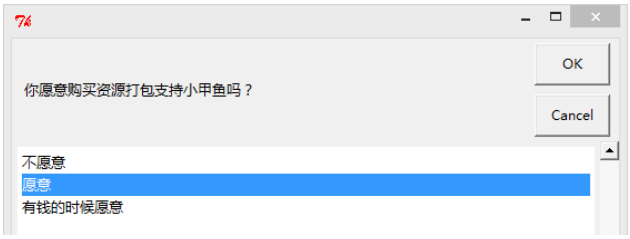
```
if ccbox():
        pass #          user chose to continue
else:
        return #        user chose to cancel
```

8 使用关键字参数调用 EasyGui 的函数

调用 EasyGui 函数还可以使用关键字参数哦.(如忘了的童鞋翻出《零基础入门学习 Python》第 18 讲自行脑补)

现在假设你需要使用一个按钮组件, 但你不想指定标题参数 (第二个参数), 你仍可以使用关键字参数的方法指定 `choices` 参数 (第三个参数) 像这样:

```
>>> choices = ['愿意', '不愿意', '有钱的时候愿意']
>>> reply = choicebox('你愿意购买资源打包支持小甲鱼吗？', choices = choices)
```



9 使用按钮组件

根据需求,EasyGui 在 `buttonbox()` 上建立了一系列的函数供调用.

9.1 msgbox()

```
1 msgbox(msg='(Your␣message␣goes␣here)', title='␣', ok_button='OK', image=None, root=None)
```

`msgbox()` 显示一个消息和提供一个 “OK” 按钮, 你可以指定任意的消息和标题, 你甚至可以重写 “OK” 按钮的内容.

以下是 `msgbox()` 的实例函数:

```
1 def msgbox(msg="(Your␣message␣goes␣here)", title="", ok_button="OK"):
2         ....
```

重写 “OK” 按钮最简单的方法是使用关键字参数:

```
1 >>> msgbox("我一定要学会编程!", ok_button="加油!")
```

9.2 ccbox()

```
1 ccbox(msg='Shall␣I␣continue?', title='␣', choices=('Continue', 'Cancel'), image=None)
```

`ccbox()` 提供一个选择:`Continue` 或者 `Cancel`, 并相应的返回 1(选中`Continue`) 或者 0(选中`Cancel`). 注意`ccbox()` 是返回整型的 1 或 0, 不是布尔类型的 `True` 或 `False`. 但你仍然可以这么写:

```
if ccbox('要再来一次吗？', choices=('要啊要啊^_^', '算了吧T_T')):
        msgbox('不给玩了，再玩就玩坏了.....')
else:
        sys.exit(0)  # 记得先 import sys 哈
```

9.3 ynbox()

```
ynbox(msg='Shall␣I␣continue?', title='␣', choices=('Yes', 'No'), image=None)
```

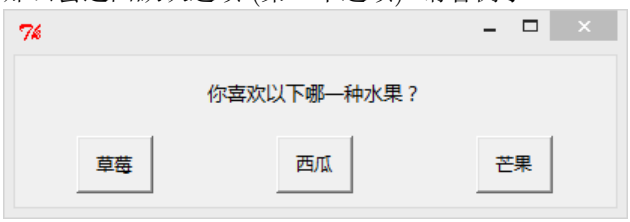
同上, 都不知作者设计这两玩意儿搞啥.

9.4 buttonbox()

```
buttonbox(msg='', title='␣', choices=('Button1', 'Button2', 'Button3'), image=None, root=None)
```

可以使用 `buttonbox()` 定义自己的一组按钮,`buttonbox()` 会显示一组你定义好的按钮.

当用户点击任意一个按钮的时候,`buttonbox()` 返回按钮的文本内容. 如果用户取消取消或者关闭窗口, 那么会返回默认选项 (第一个选项). 请看例子:



9.5 indexbox()

```
indexbox(msg='Shall␣I␣continue?', title='␣', choices=('Yes', 'No'), image=None)
```

基本跟上边一样, 区别就是当用户选择第一个按钮的时候返回序号 0, 选择第二个按钮的时候返回序号 1.

9.6 boolbox()

```
boolbox(msg='Shall␣I␣continue?', title='␣', choices=('Yes', 'No'), image=None)
```

如果第一个按钮被选中则返回 1, 否则返回 0.

10 如何在 buttonbox 里边显示图片

当你调用一个 `buttonbox` 函数 (例如 `msgbox()`, `ynbox()`, `indexbox()` 等等) 的时候, 你还可以为关键字参数 `image` 赋值, 这是设置一个.gif 格式的图像 (注意仅支持 GIF 格式哦):

```
1 buttonbox('大家说我长得帅吗？', image='turtle.gif', choices=('帅', '不帅', '!@#$%'))
```

11 为用户提供一系列选项

11.1 choicebox()

```
1 choicebox(msg='Pick something.', title='', choices=())
```

按钮组件方便提供用户一个简单的按钮选项, 但如果有很多选项, 或者选项的内容特别长的话, 更好的策略是为它们提供一个可选择的列表.

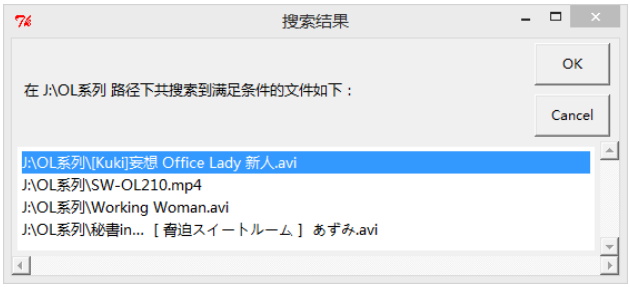
`choicebox()` 为用户提供了一个可选择的列表, 使用序列 (元祖或列表) 作为选项, 这些选项显示前会按照不区分大小写的方法排好序.

另外还可以使用键盘来选择其中一个选项 (比较纠结, 但一点儿都不重要):

1. 例如当按下键盘上的 “g” 键, 将会选中的第一个以 “g” 开头的选项. 再次按下 “g” 键, 则会选中下一个以 “g” 开头的选项. 在选中最后一个以 “g” 开头的选项的时候，再次按下 “g” 键将重新回到在列表的开头的第一个以 “g” 开头的选项.

2. 如果选项中没有以 “g” 开头的, 则会选中字符排序在 “g” 之前 (“f”) 的那个字符开头的选项.

3. 如果选项中没有字符的排序在 “g” 之前的, 那么在列表中第一个元素将会被选中.



11.2 multchoicebox()

```
1 multchoicebox(msg='Pick␣as␣many␣items␣as␣you␣like.', title='␣', choices=(), **kwargs)
```

`multchoicebox()` 函数也是提供一个可选择的列表, 与 `choicebox()` 不同的是,`multchoicebox()` 支持用户选择 0 个,1 个或者同时选择多个选项.

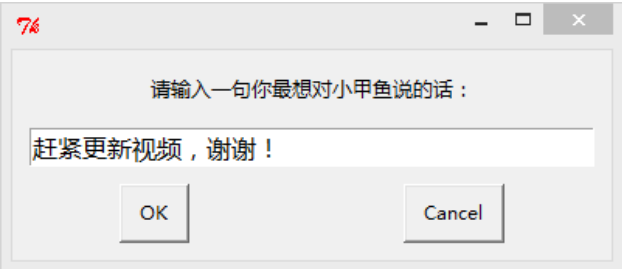
`multchoicebox()` 函数也是使用序列 (元祖或列表) 作为选项, 这些选项显示前会按照不区分大小写的方法排好序.

12 让用户输入消息

12.1 enterbox()

```
1 enterbox(msg='Enter␣something.', title='␣', default='',  strip=True, image=None, root=None)
```

`enterbox()` 为用户提供一个最简单的输入框, 返回值为用户输入的字符串. 默认返回的值会自动去除首尾的空格, 如果需要保留首尾空格的话请设置参数 `strip=False`.



12.2 integerbox()

```
integerbox(msg='', title=' ', default='', lowerbound=0, upperbound=99, image=None, root=None,
        **invalidKeywordArguments)
```

`integerbox()` 为用户提供一个简单的输入框, 用户只能输入范围内 (`lowerbound` 参数设置最小值,`upperbound` 参数设置最大值) 的整型数值, 否则会要求用户重新输入.

12.3 multenterbox()

```
multenterbox(msg='Fill in values for the fields.', title=' ', fields=(), values=())
```

`multenterbox()` 为用户提供多个简单的输入框, 要注意以下几点:

1. 如果用户输入的值比选项少的话, 则返回列表中的值用空字符串填充用户为输入的选项.
2. 如果用户输入的值比选项多的话, 则返回的列表中的值将截断为选项的数量.
3. 如果用户取消操作, 则返回域中的列表的值或者`None` 值.

实现如下图

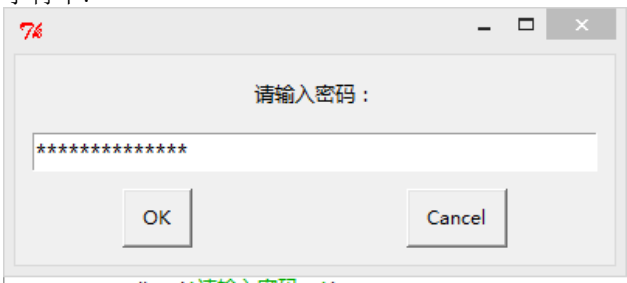
13 让用户输入密码

有时候我们需要让用户输入密码, 就是用户输入的东西看上去都是”*******”。

13.1 passwordbox()

```
1 passwordbox(msg='Enter your password.', title=' ', default='', image=None, root=None)
```

`passwordbox()` 跟 `enterbox()` 样式一样, 不同的是用户输入的内容用 “*” 显示出来, 返回用户输入的字符串:

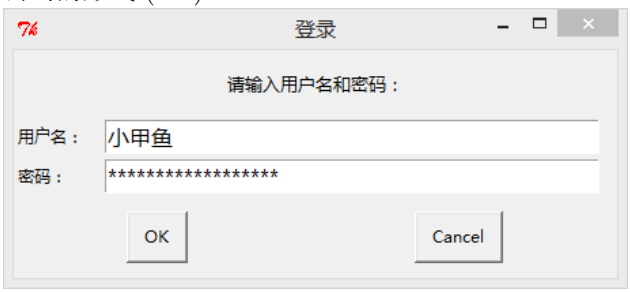



```
>>> passwordbox('请输入密码：')
'ilovefishc.com'
```

13.2 multpasswordbox()

```
1 multpasswordbox(msg='Fill␣in␣values␣for␣the␣fields.', title='␣',  fields=(),  values=())
```

`multpasswordbox()` 跟`multenterbox()` 使用相同的接口, 但当它显示的时候, 最后一个输入框显示为密码的形式 (“*”):



14 显示文本

EasyGui 还提供函数用于显示文本.

14.1 textbox()

```
1 textbox(msg='', title='␣',  text='',  codebox=0)
```

`testbox()` 函数默认会以比例字体 (参数`codebox=1` 设置为等宽字体) 来显示文本内容 (会自动换行哦), 这个函数适合用于显示一般的书面文字.

注:`text` 参数 (第三个参数) 可以是字符串类型, 列表类型, 或者元祖类型.

14.2 codebox()

```
1 codebox(msg='', title='␣', text='')
```

`codebox()` 以等宽字体显示文本内容, 相当于`textbox(codebox=1)`

注: 等宽字体很丑的, 不信你试试看

15 目录与文件

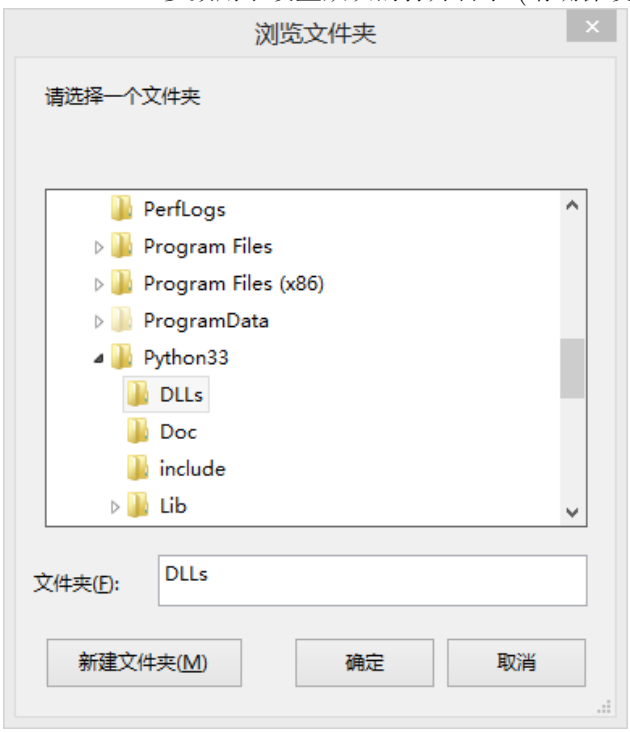
GUI 编程中一个常见的场景是要求用户输入目录及文件名,EasyGui 提供了一些基本函数让用户来浏览文件系统, 选择一个目录或文件.

15.1 diropenbox()

```
1 diropenbox(msg=None, title=None, default=None)
```

`diropenbox()` 函数用于提供一个对话框, 返回用户选择的目录名 (带完整路径哦), 如果用户选择“`Cancel`”则返回`None`.

`default` 参数用于设置默认的打开目录 (请确保设置的目录已存在).



15.2 fileopenbox()

```
1 fileopenbox(msg=None, title=None, default='*', filetypes=None)
```

`fileopenbox()` 函数用于提供一个对话框, 返回用户选择的文件名 (带完整路径哦), 如果用户选择 "`Cancel`" 则返回 `None`.

关于 `default` 参数的设置方法:

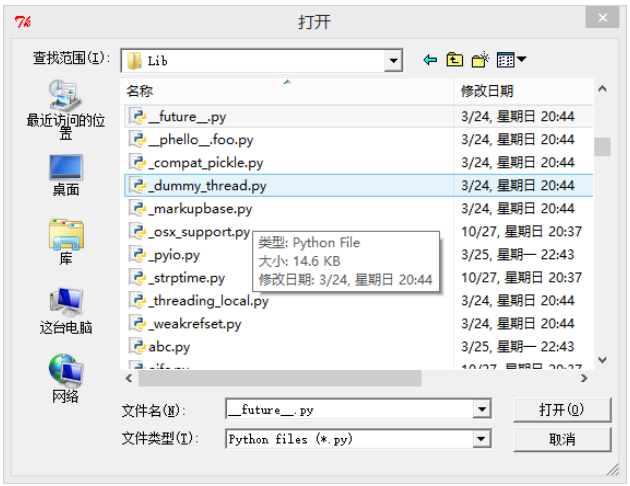
1. `default` 参数指定一个默认路径, 通常包含一个或多个通配符.
2. 如果设置了 `default` 参数,`fileopenbox()` 显示默认的文件路径和格式.
3. `default` 默认的参数是 '*', 即匹配所有格式的文件.

例如:

1. `default="c:/fishc/*.py"` 即显示 `C:\fishc` 文件夹下所有的 Python 文件.
2. `default="c:/fishc/test*.py"` 即显示`C:\fishc` 文件夹下所有的名字以 `test` 开头的 `Python` 文件.

关于 `filetypes` 参数的设置方法:

1. 可以是包含文件掩码的字符串列表, 例如:`filetypes = ["*.txt"]`
2. 可以是字符串列表, 列表的最后一项字符串是文件类型的描述, 例如:
   `filetypes = ["*.css", ["*.htm", "*.html", "HTML files"]]`



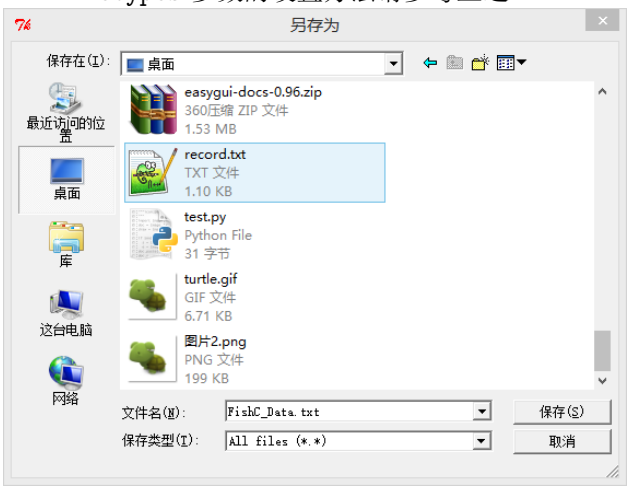
15.3 filesavebox()

```
1 filesavebox(msg=None, title=None, default='', filetypes=None)
```

`filesavebox()` 函数提供一个对话框, 让用于选择文件需要保存的路径 (带完整路径哦), 如果用户选择 "`Cancel`" 则返回 `None`.

`default` 参数应该包含一个文件名 (例如当前需要保存的文件名), 当然你也可以设置为空的, 或者包含一个文件格式掩码的通配符.

`filetypes` 参数的设置方法请参考上边.



16 记住用户的设置

16.1 EgStore

GUI 编程中一个常见的场景就是要求用户设置一下参数, 然后保存下来, 以便下次用户使用你的程序的时候可以记住他的设置.

为了实现对用户的设置进行存储和恢复这一过程,EasyGui 提供了一个叫做 `EgStore` 的类. 为了记住某些设置, 你的应用程序必须定义一个类 (暂时称之为 "设置" 类, 尽管你随意地使用你想要的名称设置它) 继承自 `EgStore` 类.

然后你的应用程序必须创建一个该类的对象 (暂时称之为 "设置" 对象)

设置类的构造函数 (`__init__` 方法) 必须初始化所有的你想要它所记住的那些值.

一旦你这样做了, 你就可以在 "设置" 对象中通过设定值去实例化变量, 从而很简单地记住设置. 之后使用 `settings.store()` 方法在硬盘上持久化设置对象.

下面是创建一个 "设置" 类的例子:

```
#-----------------------------------------------------------------------
# create "settings", a persistent Settings object
# Note that the "filename" argument is required.
# The directory for the persistent file must already exist.
#-----------------------------------------------------------------------
settingsFilename = os.path.join("C:", "FishCApp", "settings.txt") # Windows example
settings = Settings(settingsFilename)
```

下面是使用 “设置” 对象的例子:

```
# we initialize the "user" and "server" variables
# In a real application, we'd probably have the user enter them via enterbox
user =    "奥巴马"
server = "白宫"

# we save the variables as attributes of the "settings" object
settings.userId = user
settings.targetServer = server
settings.store()  #    persist the settings

# run code that gets a new value for userId
# then persist the settings with the new value
user =    "小甲鱼"
settings.userId = user
settings.store()
```

17 捕获异常

17.1 exceptionbox()

使用 EasyGui 编写 GUI 程序, 有时候难免会产生异常. 当然这取决于你如何运行你的应用程序, 当你的应用程序崩溃的时候, 堆栈追踪可能会被抛出, 或者被写入到 `stdout` 标准输出函数中.

EasyGui 通过 `exceptionbox()` 函数提供了更好的方式去处理异常, 异常出现的时候,`exceptionbox()` 会显示堆栈追踪在一个`codebox()` 中并且允许你做进一步的处理.

`exceptionbox()` 很容易使用, 下面是一个例子:

```
1 try:
2         print('I Love FishC.com!')
3         int('FISHC') # 这里会产生异常
4 except:
5         exceptionbox()
```